A PROOF OF P = NP USING MERKLE TREES

VICTOR PORTON

Abstract. My proof that P = NP.

I denote $s(X)$ the size of data X (in bits). I denote execution time of an algorithm A as $t(A, X)$.

Assume P $\neq$ NP for hashes to work.

We will need that there is a polynomial-time algorithm without hash collisions (in P $\neq$ NP assumption), because we use an algorithm based on hashing. FSB [1] is such a function.

Using a Merkle tree technology similar to one of the Cartesi [2] crypto (but with an infinite memory words count, a true random number generator, and the size of hashes associated with nodes growing logarithmically regarding the input size; because memory is infinite, we could split memory into chunks not by halves but by interleaving fragments), the size of the data and time to be used to verify the result of a decision algorithm X non-deterministically is $\log t(A, X) \cdot \log t(A, X) = 2 \log t(A, X)$. (The second multiplier is because the hash-size grows with input data size logarithmically.) We can assume that it is asymptotically $\log t(A, X)$ (because adding $s(X)$ does not matter for complexity class such problems as SAT).

Take some algorithm in NP. It's execution time is bounded by $\text{const} \cdot 2^{s(X)}$ (X is input data size).

Therefore it can be non-deterministically verified in $\log(\text{const} \cdot 2^{s(X)}) \sim s(X)$ time.

A decision problem f can be made in the same into both directions (up to a polynomial) complexity bijective function $x \mapsto (x, f(x), h(A, x))$ where $h(A, x)$ is the verifier (as in Cartesi) of execution of algorithm A of the suitable complexity class for our problem.

To have the domain and image of this decision problem well-defined and well-described, we can take f to be SAT.

So, an algorithm that computes a bijective function, which is exponential-time in both directions, is (for every bit of output) in NP (in both directions), because it is non-deterministic polynomial time (in both directions).

Therefore, it is polynomial-time (in both directions).

P = NP (the proof didn't produce an efficient algorithm).

Note that now we do know a polynomial (but not very efficient) NP-complete algorithm.

References

[1] Wikipedia contributors, "Security of cryptographic hash functions," Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/w/index.php?title=Security_of_cryptographic_hash_functions&oldid=1023078613 (accessed June 28, 2021).

[2] Teixeira Augusto, and Diego Nehab. "The Core of Cartesi." Cartesi.io: Cartesi - Smart Contracts Taken to the Next Level. Accessed June 27, 2021. https://cartesi.io/cartesi_whitepaper.pdf.

Email address: porton@narod.ru